7. Maze 3D

Program Name: Maze3D.java   Input File: maze3d.dat

NASA is testing a new drone to navigate through densely packed asteroid fields. Because of your world renowned programming skills, NASA has chosen you to write an algorithm that will find the optimal path through any asteroid field so they can make sure their drone is taking the shortest path, thus saving fuel and money. All the simulations testing your algorithm will take place in a 3-dimensional asteroid field which will contain one start point, one end point, empty spaces, asteroids and small debris. The new drone is unique in that it has a 3 shot laser cannon that can blast small debris out of its way.

The following characters will be used to describe the asteroid field for the simulations:

- `.` - Denotes empty space
- `#` - Denotes an impassible asteroid
- `*` - Denotes small debris that can be blasted away with the laser
- `S` - Denotes the start position of the drone
- `E` - Denotes the exit cell

The goal of your algorithm is to find the shortest distance through the asteroid field from the Start `S` to the Exit `E`. The drone cannot move into cells with asteroids, but it can move into cells with debris as long as it has enough laser shots left to clear the debris first. Remember, the drone only has 3 shots, so use them wisely! The drone can only move up, down, left, right, forward and back; it cannot move diagonally in any direction.

Input

The first line of input will contain a single integer `n` that indicates the number of simulations to follow. For each simulation:

- The first line contains three positive integers in the form `r c f`, where `r` is the number of rows, `c` is the number of columns, and `f` is the number of vertical "floors", or height, of the 3-dimensional area.
- Then, for each floor `f`, there will be `r` lines of input, with `c` characters in each, denoting what populates each cell of the three dimensional grid as described above.
- There will be no spaces on any of the lines, nor will there be an empty line between floors.

Output

For each simulation you will output `# MOVES` where `#` denotes the number of moves along the shortest path between the `S` and `E`, or `STUCK` if the drone cannot get to `E`.

Example Input File

```
1
4 4 4
S..#
#..#
#..#
####
.###
...#
...#
####
.###
..##
#..#
#..#
*.*.
###.
###.
###E
```

Example Output to Screen

```
9 MOVES
```